1 De-anonymizing Anonymous Data

1.1 The Infamous Netflix Dataset

We start with some examples of privacy failures and de-anonymizations of anonymized data. There exists an infamous example of the Netflix dataset which was released by Netflix between 2006 and 2009 for the Netflix Prize competition which gave a one million dollar grand prize to researchers who can improve their recommendation algorithm. They provide as training and test datasets an anonymized version of their Netflix rating data. Each datapoint consisted of anonymized userID, movieID, rating, and date. However, Narayanan and Shmatikov [NS08] showed an attack that de-anonymized a significant portion of the anonymous Netflix dataset. They cross-referenced the anonymized Netflix data with the public ratings given on IMDB by using the time of posting as reference; de-anonymize by matching similar ratings to the same movie at around the same timestamp. They successfully de-anonymized a large portion of the data.

2 Reconstruction Attack

Reconstruction attacks seek to reconstruct individual datapoints from aggregate statistics. Today, we talked about the reconstruction attack lower bound of Dinur and Nissim [DN03] (which came before the definition for differential privacy!)

Suppose you have a database where rows contain datapoints and columns contain features. Some of these features are private, such as whether someone has a certain disease. Suppose the private feature is represented by a private vector d of dimension n. The vector contains a i in the i-th location if person i has the disease and 0 otherwise. You are allowed to ask queries of the following form: "How many rows satisfy CONDITION and HAS_DISEASE = 1?" The CONDITION could be, for example, BIRTH_MONTH=February.

We can also represent queries using a vector $S \in \{0,1\}^n$ where 1 indicate indices which satisfy the query and 0 for indices that do not. These are called *subset queries*. We can compute the answer to a query via the inner product between d and S: $A(S) = d \cdot S$ (dot product). Hence, these specific queries are also known as *inner product queries*. Suppose we want to keep the data in d private, then we return a $r(S)$ which is not equal to $A(S)$ since if we return $A(S)$ exactly, one can just query each individual one-by-one and get the precise information on each individual. Thus, we need to output a "noisy" $r(S)$ where $|r(S) - A(S)| \leq E$ if the error bound for $r(S)$ is bounded by E. Here, $r(S)$ doesn't necessarily need to be randomly distributed.

Definition 2.1 (Blatantly Non-Private)**.** An algorithm $\mathcal{A}$ is ***blatantly non-private*** if an adversary can construct a database $c \in \{0,1\}^n$ such that it matches the true database d in all but $o(n)$ entries.

Theorem 2.2. *If an adversary is allowed 2^n subset queries, and the curator adds noise with some bound E, then based on the results, the adversary can reconstruct the database in all but $4E$ positions.*

Proof. The adversary runs the following algorithm. They ask for *all* but 2^n subsets of $[n]$ and output any database which produces *consistent* answers. Precisely speaking, for each possible candidate database $c \in \{0,1\}^n$, if there exists query set S such that $|\sum_{i \in S} c_i - r(S)| > E$, then rule out c. The private database d would not be ruled out so there exists at least one database that would not be ruled out.

Now, we compute the maximum error of the databases. Let $I_0 = \{i \mid d[i] = 0\}$ be the set of bits in d that are 0 and $I_1 = \{i \mid d[i] = 1\}$ be the set of bits in d that are 1. Of course, the adversary does not know I_0 and I_1, but we use them for the sake of analysis. By the adversary's strategy, it holds that $|\sum_{i \in I_0} c_i - r(I_0)| \leq E$. Furthermore, we guarantee that the curator can add at most E noise so $|\sum_{i \in I_0} d_i - r(I_0)| \leq E$. By the triangle inequality, c and d differ by at most $2E$ entries for I_0. A symmetric argument states that c and d differ by at most $2E$ entries for I_1. Thus, they differ by $4E$ overall. □

The following theorem by Dinur and Nissim [DN03] shows a stronger result.

Theorem 2.3. *If the adversary is allowed to ask $\Theta(n)$ subset queries, and the curator adds noise with some bound $E = O(\sqrt{n})$, then any algorithm that adds E noise is blatantly non-private.*

3 Privacy Definitions

When discussing differential privacy, there are several considerations we need to take into account when determining the setting for which privacy is applied.

Trust Model Who gets access to the private data?

- *Trusted curator:* If we have a ***trusted curator***, then we are operating in the ***central model of differential privacy***. Given n individuals holding private values, $X_1, \ldots, X_n$, each sends their private data to the trusted curator and no one else. The curator then runs some differentially private (DP) algorithm (also known as a DP *mechanism*) $\mathcal{M}(X_1, \ldots, X_n)$ on the private data and outputs privatized answers.
- *Untrusted curator:* If we have an ***untrusted curator***, then we are operating in the ***local model of differential privacy***. All individuals run their private values X_i through a local DP mechanism $\mathcal{M}(X_i)$ which outputs private answers to give to the curator. Very weak model of trust; individuals trust no one with their data!
- *Other trust models in between:* ***Shuffle model*** has a trusted *shuffler* that produces a uniformly-at-random permutation on the privatized data that is sent to it. I won't be covering these in-between models in this lecture.

Neighboring Inputs Privacy is guaranteed for pairs of ***neighboring*** inputs. Given a private mechanism, $\mathcal{M} : \mathcal{X}^n \to \mathcal{Y}$. Neighboring inputs are two datasets on n parties, $X, X' \in \mathcal{X}^n$, which differ in exactly one entry. In other words, X neighbors X' when there exists exactly one $j \in [n]$ where $X[j] \neq X'[j]$ and $X[i] = X'[i]$ for all other $i \neq j \in [n]$. We denote neighboring datasets by $X \sim X'$.

Other types of neighboring inputs: For graphs, the common notions are ***edge-neighboring*** and ***node-neighboring*** inputs. ***Edge-neighboring*** graphs are graphs that differ by exactly one edge. And ***node-neighboring*** graphs are graphs that differ in exactly one node and adjacent edges.

Formal Definitions Now, I will formally give the definitions for the ***(central) differential privacy (DP)*** model and the ***local differential privacy (LDP)*** model.

Definition 3.1 ((Central) Differential Privacy (DP) Model [DMNS06])**.** Let $\varepsilon > 0$ and $\delta \in [0, 1)$. A randomized algorithm $\mathcal{A}$ is (ε, δ)*-differentially private (DP)* (with respect to the neighbor relation on the universe of the datasets) if for all events S in the output space of $\mathcal{A}$ and all neighboring datasets X and X',

$$\Pr[\mathcal{A}(X) \in S] \leq \exp(\varepsilon) \cdot \Pr[\mathcal{A}(X') \in S] + \delta.$$

In Definition 3.1, when $\delta = 0$, the algorithm is ε-differentially private (sometimes called *purely* differentially private). Otherwise, when $\delta > 0$, the algorithm is *approximately* differentially private. I provide the definitions in terms of both ε and δ but for simplicity and due to time constraints, we will discuss results in terms of *pure* DP (when $\delta = 0$) for the remainder of this lecture.

The LDP model is defined in terms of a ***local randomizer***. This definition looks very similar to the definition of (central) DP, and it is very similar, but with one key distinction. The privacy guarantee is over the *private data held by each individual party* as opposed to over all of the private data held by all of the parties.

Definition 3.2 (Local Randomizer [KLN+11])**.** Let $\varepsilon > 0$ and $\delta \in [0, 1)$. A ***local randomizer*** $\mathcal{R} : \mathcal{X} \to \mathcal{Z}$ is an (ε, δ)*-differentially private (DP)* algorithm where all pairs of neighboring inputs are given by $x, x' \in \mathcal{X}$.

It is easiest to define local privacy in terms of the *composition* of multiple calls to local randomizers. ***Composition*** is used to show that running multiple independent DP mechanisms on the dataset results in a differentially private algorithm (with proportionally larger privacy parameters).

Lemma 3.3 (Basic Composition)**.** *Let $M_1, M_2, \cdots, M_k : \mathcal{X}^n \to \mathcal{Y}$ be randomized algorithms. Suppose M_j is ε_j-DP for each $j \in [k]$. Define $M : \mathcal{X}^n \to \mathcal{Y}^k$ by $M(x) = (M_1(x), M_2(x), \cdots, M_k(x))$, where each algorithm is run independently. Then, M is ε-DP where $\varepsilon = \sum_{j=1}^{k} \varepsilon_j$.*

Proof. Fix neighboring datasets $x, x' \in \mathcal{X}^n$ and output $y_j \in Y^k$ for each $j \in [k]$. By definition, we have for each j, $e^{-\varepsilon_j} \leq \frac{\mathsf{Pr}[M_j(x)=y_j]}{\mathsf{Pr}[M_j(x')=y_j]} \leq e^{\varepsilon_j}$. Then, by independence, we have

$$\prod_{j=1}^{k} e^{-\varepsilon_j} \leq \prod_{j=1}^{k} \frac{\mathsf{Pr}[M_j(x) = y_j]}{\mathsf{Pr}[M_j(x') = y_j]} \leq \prod_{j=1}^{k} e^{\varepsilon_j} = e^{\sum_{j=1}^{k} \varepsilon_j} = e^{\varepsilon}.$$

□

Definition 3.4 (Local Differential Privacy (Informal) [KLN+11])**.** Given an algorithm $\mathcal{A}$ that calls a set of local randomizers $[\mathcal{R}_{1,1}, \ldots, \mathcal{R}_{v,i}, \ldots, \mathcal{R}_{n,r}]$ for each party in $[n]$, algorithm $\mathcal{A}$ is $(\varepsilon_{v,i}, \delta_{v,i})$-locally differentially private if for every $v \in [n]$, the composition of all calls to local randomizers $\mathcal{R}_{v,i}$ for all i is (ε, δ)-differentially private.

[DLR+22] (Definition 2.9) gives a more general definition than the above for local privacy on graphs via a transcript-based definition.

3.1 More About Composition Theorems

Composition allows us to show privacy is preserved under ***group privacy***. In other words, if two neighboring datasets differ in k entries, then any ε-DP mechanism run on this dataset is $(k \cdot \varepsilon)$-DP. Furthermore, we can show that composition holds adaptively also.

Theorem 3.5 (Adaptive Composition Theorem [DMNS06, DL09, DRV10])**.** *A sequence of DP algorithms, $(\mathcal{A}_1, \ldots, \mathcal{A}_k)$, with privacy parameters $(\varepsilon_1, \ldots, \varepsilon_k)$ form at worst an $(\varepsilon_1 + \cdots + \varepsilon_k)$-DP algorithm under* adaptive composition *(where the adversary can adaptively select algorithms after seeing the output of previous algorithms).*

Proof. Let M be an ε-DP mechanism and $\boldsymbol{y} = (y_1, \ldots, y_{k-1})$ be a set of k outputs for adaptively chosen adjacent inputs $\boldsymbol{x} = (x_0, x_1, \ldots, x_{k-1})$ and $\boldsymbol{x'} = (x'_0, x'_1, \ldots, x'_{k-1})$ to the mechanism where $y_i \in \mathcal{Y}$ for every $i \in [k-1]$. Let the randomness over M be a discrete probability distribution. Below, we abuse notation and let $M(\boldsymbol{x})$ denote the set of outputs obtained by running M on adaptive inputs $\boldsymbol{x}$. We then have:

$$\begin{aligned}\frac{\mathsf{Pr}[M(\boldsymbol{x}) = \boldsymbol{y}]}{\mathsf{Pr}[M(\boldsymbol{x'}) = \boldsymbol{y}]} &= \left(\frac{\mathsf{Pr}[M(x_0) = y_1]}{\mathsf{Pr}[M(x'_0) = y_1]}\right) \cdot \prod_{i=1}^{k-1} \frac{\mathsf{Pr}[M(x_i) = y_i | y_1, \ldots, y_{i-1}]}{\mathsf{Pr}[M(x'_i) = y_i | y_1, \ldots, y_{i-1}]} \\ &\leq \prod_{i=1}^{k} \exp(\varepsilon) = \exp(k\varepsilon).\end{aligned}$$

The second inequality follows since M is ε-DP, each pair of inputs are adjacent, and previous outputs are public information. One can also prove this lemma for continuous distributions. □

3.2 Post-Processing

The last characteristic of differential privacy I want to mention is that privacy is preserved after *post-processing*. In other words, any function that is performed on the output of an (ε, δ)-DP algorithm is also (ε, δ)-DP.

Lemma 3.6 (Privacy is Preserved after Post-Processing)**.** *Let $M : \mathcal{X}^n \to \mathcal{Y}$ be a (ε, δ)-differentially private algorithm. Let $f : \mathcal{Y} \to \mathcal{Y}'$ be an arbitrary mapping. Then, $f \circ M : \mathcal{X}^n \to \mathcal{Y}'$ is (ε, δ)-differentially private.*

3.3 Some Notes about Differential Privacy

There are a number of characteristics about differential privacy that are useful to consider but may not be immediately clear upon first hearing about these definitions.

- Differential privacy is not binary but quantitative. It is not the case that you either *have* or *don't have* privacy; rather you can have *greater* privacy or *less* privacy. Smaller ε corresponds with stronger privacy guarantees and larger ε leads to weaker privacy guarantees.
- ε should be small, ideally, $\varepsilon \in (0, 1]$. Anything with $\varepsilon > 5$ should be viewed with some skepticism.
- Privacy is guaranteed in the *worst-case* with approximation factors (or utility) guaranteed with high probability. Worst-case means that privacy is guaranteed for *all* possible datasets.
- The choice of e^ε in the multiplicative factor is useful for composition. For small ε, it holds that $e^\varepsilon \approx (1 + \varepsilon)$ by Taylor expansion.
- The choice of e results from the early mechanisms like the Laplace mechanism whose PDF is given in terms of e.
- Any *non-trivial* DP algorithm must be randomized. Trivial means any algorithm that outputs something independent of the dataset, i.e. any algorithm that outputs 0 regardless of input.

For the rest of this lecture, we will mainly focus on central DP except when LDP can be easily obtained, in which case, I will comment on such cases. The following content will be for discrete distributions.

4 Privacy Mechanisms

There are standard mechanisms for obtaining privacy in a variety of cases.

4.1 Randomized Response

As the name suggests, this mechanism is for the case when you randomly decide whether to provide a response. The nice thing about Randomized Response is that it provides local privacy (in addition to central privacy)!

Randomized response is a standard and useful way to learn about the statistics of a population when the individual random bits held by each person is sensitive. Suppose you are surveying your university about who has gotten COVID within the past month and what fraction of the university has gotten COVID. Each individual has a bit $X_i \in \{0, 1\}$ and would like no one else to learn the true value of X_i; thus, each individual sends to the coordinator a bit Y_i that depends on X_i and some randomness generated independently by each individual. Using the Y_i's, the curator computes the statistic, $\tilde{p} = \frac{1}{n} \sum_{i=1}^{n} Y_i$ which is some estimate of the

the true value $p = \frac{1}{n}\sum_{i=1}^{n} X_i$. There are two approaches: one which gets no privacy and one which gets complete privacy. An exercise for the reader: What are these two approaches?

Now, we reach a compromise between the two. Suppose we have some $\epsilon \in (0, 1/2]$, and we send Y_i as follows:

$$Y_i = \begin{cases} X_i & \text{with probability } 1/2 + \varepsilon \\ 1 - X_i & \text{with probability } 1/2 - \varepsilon. \end{cases}$$

What follows are then two natural questions: is this mechanism private? And what error do we obtain from the mechanism?

The privacy of this mechanism directly follows from composition and post-processing. Suppose our algorithm $M(X_1, \ldots, X_n) = (Y_1, \ldots, Y_n)$ is randomized response. We show that $(Y_1, \ldots, Y_n)$ is differentially private, thus satisfying local privacy. We show our calculation for (central) privacy, although local privacy implies central privacy in this setting.

We fix the outputs to be $a = (a_1, \ldots, a_n)$. Then, $\mathsf{Pr}[M(X) = a] = \prod_{i=1}^{n} \mathsf{Pr}[Y_i = a_i]$. Suppose that neighboring datasets X and X' differ in coordinate j. Then,

$$\frac{\mathsf{Pr}[M(X) = a]}{\mathsf{Pr}[M(X') = a]} = \frac{\prod_{i=1}^{n} \mathsf{Pr}[Y_i = a_i]}{\prod_{i=1}^{n} \mathsf{Pr}[Y_i' = a_i]} = \frac{\mathsf{Pr}[Y_j = a_j]}{Y_j' = a_j} \leq \frac{1/2 + \varepsilon}{1/2 - \varepsilon} \leq e^{O(\varepsilon)},$$

for small ε.

Now, we compute the accuracy of the mechanism. First, we compute the expectation to be

$$\mathbb{E}[Y_i] = X_i \cdot (1/2 + \varepsilon) + (1 - X_i) \cdot (1/2 - \varepsilon) = 2\varepsilon \cdot X_i + 1/2 - \varepsilon.$$

We want the expectation to be equal to X_i, so we have the following natural estimator where $\mathbb{E}[\tilde{p}] = p$ (using linearity of expectations):

$$\tilde{p} = \frac{1}{n}\sum_{i=1}^{n} \left[\frac{1}{2\varepsilon}(Y_i - 1/2 + \varepsilon)\right].$$

Now, we analyze the variance of $\tilde{p}$:

$$\mathbf{Var}[\tilde{p}] = \mathbf{Var}\left[\frac{1}{n}\sum_{i=1}^{n}\left[\frac{1}{2\varepsilon}(Y_i - 1/2 + \varepsilon)\right]\right] = \frac{1}{4\varepsilon^2 n^2}\sum_{i=1}^{n}\mathbf{Var}[Y_i] \leq \frac{1}{16\varepsilon^2 \cdot n},$$

where $\mathbf{Var}[Y_i]$ is the variance of a Bernoulli random variable with parameter $1/2 + \varepsilon$ which is at most $1/4$. We can now use Chebyshev or Chernoff to show concentration around $|\tilde{p} - p| \leq O\left(\frac{1}{\varepsilon \cdot \sqrt{n}}\right)$. This essentially means that when you require more privacy, you also need more samples.

There exists a mechanism where we can obtain better error and that is the ***geometric mechanism***. The geometric mechanism is the discrete version of the Laplace mechanism and has the advantage of being easier to work with for problems where the outputs are discrete, is (potentially) more practical, and is *universally utility-maximizing* [GRS09].

4.2 Geometric Mechanism

The Geometric Mechanism, like its sister mechanism, the Laplace Mechanism, is useful for numerical queries. In order to use this mechanism, we have to define another characteristic of neighboring datasets, which is the ***sensitivity*** of a function f. The sensitivity of function f, denoted by Δf, is the maximum ℓ_1 distance between the outputs of f on two neighboring datasets $X \sim X'$:

$$\Delta f = \max_{X \sim X'} ||f(X) - f(X')||_1.$$

Let's consider the simple example above when we want to compute the fraction of individuals who have gotten COVID within the past month. What is the sensitivity of that function? (Answer is given later.)

Now, we are ready to introduce the Geometric Mechanism. The Geometric Mechanism is defined in terms of the ***symmetric geometric distribution*** which is defined as follows:

Definition 4.1 (Symmetric Geometric Distribution [BV18, SCR+11])**.** The ***symmetric geometric distribution***, denoted $\mathsf{Geom}(b)$, with input parameter $b \in (0, 1)$, takes integer values i where the probability mass function at i is $\frac{e^b-1}{e^b+1} \cdot e^{-|i| \cdot b}$.

The symmetric geometric distribution outputs a positive, negative or 0 integer each with mean centered at 0, smaller ε results in the a "flatter" distribution. Using this distribution, we can now define the ***geometric mechanism***.

Definition 4.2 (Geometric Mechanism [BV18, CSS11, DMNS06, DNPR10])**.** Given any function $f : \mathcal{D} \to \mathbb{Z}^d$, where $\mathcal{D}$ is the domain of f and Δf is the ℓ_1-sensitivity of f, the geometric mechanism is defined as $M(x, f(\cdot), \varepsilon) = f(x) + (Y_1, \ldots, Y_d)$, where $Y_i \sim \mathsf{Geom}(\varepsilon/\Delta f)$ are i.i.d. random variables drawn from $\mathsf{Geom}(\varepsilon/\Delta f)$ and x is a data set.

In our example above, the sensitivity of $f = p \cdot n$ is $1/n \cdot n = 1$. Hence, we apply the geometric mechanism with $\Delta f = 1$ and $\mathsf{Geom}(\varepsilon)$. Thus, we compute $\tilde{p} \cdot n = f(X) + \mathsf{Geom}(\varepsilon)$. Recall, we previously defined $\tilde{p}$ so that $\mathbb{E}[\tilde{p} \cdot n] = p \cdot n$ (since n is fixed). It follows that $\mathbb{E}[f(X) + \mathsf{Geom}(\varepsilon)] = f(X)$ by linearity of expectations and since $\exp[\mathsf{Geom}(\varepsilon)] = 0$. Now we compute the variance:

$$\mathbf{Var}\left[f(X) + \mathsf{Geom}(\varepsilon)\right] = \mathbf{Var}[\mathsf{Geom}(\varepsilon)] = O\left(\frac{1}{\varepsilon^2}\right).$$

Thus, $\mathbf{Var}[\tilde{p}] = \mathbf{Var}\left[\frac{f(X)+\mathsf{Geom}(\varepsilon)}{n}\right] = O\left(\frac{1}{\varepsilon^2 \cdot n^2}\right)$. Again, by Chernoff or Chebyshev, we can show concentration around the error to be $|\tilde{p} - p| = O\left(\frac{1}{\varepsilon n}\right)$. *Important note:* The geometric mechanism as used here is *not locally private*. Local privacy would result in *much* more noise. I'll leave the privacy proof of the geometric mechanism as an exercise for you! (But proof is given in the appendix.)

There are several other important mechanisms that I do not have time to cover in this lecture: the *Laplace mechanism*, the *Gaussian Mechanism*, and the *Exponential Mechanism*. I encourage you to read the linked lecture notes to learn about these two mechanisms.

5 Sources Used for Lecture Notes

I've used a number of sources, in addition to the papers cited in the References. I've listed these sources below and I encourage you to look at these wonderful resources (and more that I haven't listed) to learn more about differential privacy.

1. Gautam Kamath's course notes (used extensively in this lecture): Algorithms for Private Data Analysis.
2. Thomas Steinke's chapter in *Differential Privacy for Artificial Intelligence Applications*: Composition of Differential Privacy & Privacy Amplification by Subsampling.
3. Adam Smith's blog: Oddly Shaped Pegs.
4. Cynthia Dwork and Aaron Roth's privacy book: The Algorithmic Foundations of Differential Privacy.
5. The Differential Privacy blog: DifferentialPrivacy.org.

References

[BV18] Victor Balcer and Salil P. Vadhan. Differential privacy on finite computers. In *9th Innovations in Theoretical Computer Science Conference, ITCS 2018, January 11-14, 2018, Cambridge, MA, USA*, pages 43:1–43:21, 2018.

[CSS11] T.-H. Hubert Chan, Elaine Shi, and Dawn Song. Private and continual release of statistics. *ACM Trans. Inf. Syst. Secur.*, 14(3), November 2011.

[DL09] Cynthia Dwork and Jing Lei. Differential privacy and robust statistics. In *Proceedings of the Forty-First Annual ACM Symposium on Theory of Computing*, STOC '09, page 371–380, New York, NY, USA, 2009. Association for Computing Machinery.

[DLR+22] Laxman Dhulipala, Quanquan C. Liu, Sofya Raskhodnikova, Jessica Shi, Julian Shun, and Shangdi Yu. Differential privacy from locally adjustable graph algorithms: k-core decomposition, low out-degree ordering, and densest subgraphs. In *63rd IEEE Annual Symposium on Foundations of Computer Science, FOCS 2022, Denver, CO, USA, October 31 - November 3, 2022*, pages 754–765. IEEE, 2022.

[DMNS06] Cynthia Dwork, Frank McSherry, Kobbi Nissim, and Adam Smith. Calibrating noise to sensitivity in private data analysis. In *Proceedings of the Third Conference on Theory of Cryptography*, TCC'06, page 265–284, Berlin, Heidelberg, 2006. Springer-Verlag.

[DN03] Irit Dinur and Kobbi Nissim. Revealing information while preserving privacy. In *Proceedings of the twenty-second ACM SIGMOD-SIGACT-SIGART symposium on Principles of database systems*, pages 202–210, 2003.

[DNPR10] Cynthia Dwork, Moni Naor, Toniann Pitassi, and Guy N. Rothblum. Differential privacy under continual observation. In *Proceedings of the Forty-Second ACM Symposium on Theory of Computing*, STOC '10, pages 715–724, New York, NY, USA, 2010. Association for Computing Machinery.

[DRV10] Cynthia Dwork, Guy N. Rothblum, and Salil Vadhan. Boosting and differential privacy. In *Proceedings of the 2010 IEEE 51st Annual Symposium on Foundations of Computer Science*, FOCS '10, page 51–60, USA, 2010. IEEE Computer Society.

[GRS09] Arpita Ghosh, Tim Roughgarden, and Mukund Sundararajan. Universally utility-maximizing privacy mechanisms. In *Proceedings of the forty-first annual ACM symposium on Theory of computing*, pages 351–360, 2009.

[KLN+11] Shiva Prasad Kasiviswanathan, Homin K Lee, Kobbi Nissim, Sofya Raskhodnikova, and Adam Smith. What can we learn privately? *SIAM Journal on Computing*, 40(3):793–826, 2011.

[NS08] Arvind Narayanan and Vitaly Shmatikov. Robust de-anonymization of large sparse datasets. In *2008 IEEE Symposium on Security and Privacy (sp 2008)*, pages 111–125. IEEE, 2008.

[SCR+11] Elaine Shi, T.-H. Hubert Chan, Eleanor Gilbert Rieffel, Richard Chow, and Dawn Song. Privacy-preserving aggregation of time-series data. In *Proceedings of the Network and Distributed System Security Symposium, NDSS 2011, San Diego, California, USA, 6th February - 9th February 2011*, 2011.